

# 《代码管理：git使用规范》

作者：王琦

时间：2016年4月24日

摘要：本文先给出了**代码管理工具**的下载地址，再描述了项目的**创建方式**，之后以**分支管理**、**代码审查**（自我审查与他人审查）为例，指出了在开发过程中应遵循的行为规范和实现方式。希望大家在开发的过程中，能遵守规范，掌握在**命令行**或图形界面下进行代码管理的方法。

## 1. 工具下载

下载[Github desktop](#)

下载[Sourcetree](#)

## 2. 项目创建

### 2.1 配置公钥

配置公钥的目的，是使得今后在进行git操作时，不用**每次**都需要进行身份验证。

配置教程可参考[配置SSH公钥](#)

### 2.2 克隆项目

```
git clone git@git.coding.net:stephenwww/suri.git
```

## 3. 项目开发

### 3.1 分支管理

在此我们先给出分支管理的**逻辑**，在本节末给出完整的git指令

#### 3.1.1 分支创建

每开发一个功能：

- 都应该基于当前最新的dev分支，去创建一个新的分支
- 该分支名应当**自解释**了当前分支负责开发的功能——给分支取一个有意义的名字**很重要**

假如你正在dev分支下，你应该使用

```
git checkout -b your_new_branch
```

去创建一个，基于dev分支的，名为your\_new\_branch的新分支。

### 3.1.2 将发生修改的文件添加到commit list

在完成当前分支开发后，需要做两件事：

- 进行代码的自我审查（在3.2.1中描述）
- 将发生修改的文件添加到commit list中，有两种方法可以实现这一点：
  - 通过`git add your_filename`指令，将与your\_filename参数匹配的文件添加到commit list中
  - 通过sourcetree软件，在unstaged files标签中，选择文件将它添加到staged files中

### 3.1.3 提交代码(commit)

通过git commit指令提交代码，注意：

- -m参数后需要添加一段提交注释
- 我们项目要求：注释应解释当前提交行为，使用英文，并以大写字母开头。

示例：

```
git commit -m "Modify CMakeList.txt to add a console when program running."
```

### 3.1.4 推送代码(rebase & push)

在推送代码之前，应确保当前分支仍基于最新的dev分支：因为在checkout和push之间，有可能别的开发者更新了dev分支：

```
git rebase dev
```

在成功rebase后，再将当前分支推送到远程仓库中：

```
git push -u origin your_new_branch
```

### 3.1.5 命令合集

开发过程应至少使用如下指令：

```
git checkout -b your_new_branch
... // Development begins
...
... // Development ends
git add ...
git commit -m "Your meaningful commit message."
git rebase dev
git push -u origin your_new_branch
```

## 3.2 代码审查

### 3.2.1 自我审查

在两个环节应进行代码的自我审查：

- 在3.1.2提交代码，进行git add指令时
- 在3.1.4推送代码，进行git rebase指令时

审查的内容包括但不限于：

- 代码是否符合C++标准规范(C++11, C++14)
- 代码是否符合项目代码规范
  - suri基本采用Google C++ Code Style
  - huacaya应尽快确立自己的代码规范（可适当参考suri）
- 代码是否可以写得更高效、更可读（简洁）

### 3.2.2 他人审查

他人审查发生在分支被push到远程仓库后。

项目成员角色暂划分为：

- Reviewer: 王琦
- Non-reviewer: 张雷、林立文

#### 3.2.2.1 Non-reviewer push

Non-reviewer A在将分支push到远程仓库后，应和reviewer B沟通，经B审查后，由B将分支merge到dev上。

#### 3.2.2.2 Reviwer push

Reviwer A将代码push到远程仓库后，应与另一位reviwer B沟通，经B审查后，由B将分支merge到dev上。

## 4. 总结

本文给出了代码**管理工具**的下载地址，描述了项目的**创建方式**，最后以**分支管理**、**代码审查**（自我审查与他人审查）为例，指出了在开发过程中应遵循的行为规范和实现方式。

希望大家在今后的开发过程中，能遵守规范，掌握在 *命令行*或图形界面下进行代码管理的方法。